Couche Application

Laurent Signac – CC-BY-SA – 24-11-22 1345 7aa51bd4a79a2b0a1e6e

Nous décrivons dans ce chapitre un certain nombre de protocoles de la couche application du modèle TCP/IP. Ces protocoles sont basés sur TCP ou sur UDP. Le nombre de protocoles qu'il serait envisageable de traiter est considérable. Nous avons donc opéré un choix sur les applications traitées.

1 Hypertext transfert protocol

Le protocole HTTP (Hypertext Transfert Protocol) est l'un des plus utilisés sur Internet car il concerne le World Wide Web. Ce protocole décrit de quelle façon un navigateur peut interroger un serveur Web, et lui demander la donnée qui correspond à une URL (*Uniform Ressource Location*). Ce protocole utilise TCP. La communication s'effectue en deux temps : le client (un navigateur par exemple) envoie une requête HTTP à un serveur, qui lui répond. Le port utilisé par le protocole HTTP est généralement le port 80.

Une requête HTTP comprend trois parties, dont seule la première est obligatoire :

1. Ligne de requête
2. Lignes d'en-tête
3. Corps de la requête

La ligne de requête est elle-même composée de trois parties : la méthode, l'URL, et la version du protocole utilisé. Dans la version 1.1 de HTTP, les méthodes sont au nombre de 8 : Options, Get, Head, Post, Put, Delete, Trace et Connect (dans HTTP 1.0, les seules méthodes sont Get, Head, et Post). La dernière version de HTTP est HTTP/2.

Options : Demande d'information au serveur.

Get : C'est la méthode la plus employée. Elle permet de récupérer le contenu d'une URL.

Head : Cette méthode ne renvoie que les en-têtes qui auraient été renvoyés par la méthode Get. Elle ne renvoie pas le contenu lui-même. Elle permet entre autres de contrôler la date et la taille du contenu d'une URL.

Post : Cette méthode permet de passer des données avec la requête. Elle est par exemple utilisée lors de la validation d'un formulaire. Une requête concernant l'URL de traitement du formulaire est faite, et le contenu du formulaire est passé avec le corps de la requête.

Put : Permet de demander la création d'une URL (ou son remplacement). Le contenu est donné dans le corps de la requête.

Delete : Demande d'effacement d'une URL.

Trace : Demande d'écho. Le serveur (ou le proxy) doit renvoyer le message reçu vers le client, afin de lui permettre de diagnostiquer un problème éventuel.

Le protocole HTTP en version 1.1 fonctionne en mode connecté (en version 1.0, le client était déconnecté à chaque requête).

1.1 Requête Get

Voici par exemple un dialogue avec un serveur Web, réalisable manuellement avec `telnet` ou le programme `PuTTY` (on y distingue 3 parties : ce qui est envoyé par le client, puis la réponse du serveur, contenant les entêtes suivis de la page Web) :

```
$ telnet 194.254.43.242 80

Trying 194.254.43.242...
Connected to 194.254.43.242.
Escape character is '^]'.
GET /demo/page.html HTTP/1.1
Host: deptinfo-ensip.univ-poitiers.fr

HTTP/1.1 200 OK
Server: nginx/1.14.0
Date: Thu, 20 Jun 2020 16:05:20 GMT
Content-Type: text/html
Content-Length: 117
Last-Modified: Wed, 03 Jun 2020 13:55:53 GMT
Connection: keep-alive
ETag: "506c43e9-75"
Accept-Ranges: bytes

<html>
<head>
</head>
<body>
<h1> La belle page !!! </h1>
Et la belle image : <img src="nfrac.png"/>
</body>
</html>
```

2 Post Office Protocol

Le protocole POP (Post Office Protocol) est actuellement utilisé en version 3. Il permet de récupérer du courrier électronique sur un serveur distant appelé serveur POP et de le rapatrier sur la machine locale (par opposition à IMAP (Internet Message Access Protocol) qui est généralement utilisé pour consulter les courriers électroniques, tout en les laissant sur le serveur distant, bien que tout ceci soit paramétrable). Ce protocole utilise TCP sur le port 110. Les requêtes sont composées d'une commande et d'un ou plusieurs arguments.

Voici le descriptif de quelques commandes POP3 :

`User <identifiant>` : indique le nom de login de l'utilisateur (connexion)

`Pass <password>` : indique le mot de passe (connexion)

`Stat` : demande le nombre de messages sur le serveur

`Retr <num>` : récupére d'un message

`Dele <num>` : supprime d'un message

`Top <num> <nb>` : affiche les premières lignes d'un message

`Quit` : ferme la connexion

Le protocole POP fonctionne en mode connecté, en ce sens qu'après l'établissement de la connexion, le client peut faire plusieurs requêtes.

Voici un exemple de dialogue entre un client et un serveur POP :

```
$ telnet pop3.bidule.fr 110

Trying 10.16.83.27...
Connected to pop3.bidule.fr.
Escape character is '^]'.
+OK <29644.1089906610@pop4-q.bidule.fr>
USER jonsnow
+OK
PASS tUrlututuChAp09ointu
+OK
STAT
+OK 6 144045
TOP 1 10
+OK 58775 octets
Return-Path: <Error_Info@hotmail.com>
Delivered-To: online.fr-jonsnow@bidule.fr
Received: (qmail 22453 invoked from network); 21 Apr 2004 11:24:09 -0000
Received: from aneuilly-109-1-3-166.w81-48.abo.wanadoo.fr (HELO error?info.com)
  by mrelay2-1.blip.fr with SMTP; 21 Apr 2004 11:24:09 -0000
From: Error_Info@hotmail.com
To: Mail@blip.fr
Subject: Invalid mail sentence length / You know nothing
Importance: Normal
X-Priority: 3 (Normal)
Message-ID: <384cb13db0bad1.a071f.qmail@hotmail.com>
MIME-Version: 1.0
...
QUIT
+OK
Connection closed by foreign host.
```

3 Simple Message Transfert Protocol

Le protocole SMTP est utilisé pour relayer les courriers électroniques de machine en machine lors de leur expédition. Le port officiel du protocole SMTP est le port 25 et c'est un protocole en mode connecté qui fonctionne avec TCP. Les principales commandes SMTP sont : Ehlo, Mail from, Rcpt to et Data. Voici un exemple de transaction SMTP :

```
$ telnet smtp.bla.fr 25

Trying 10.16.66.83...
Connected to smtp.bla.fr.
Escape character is '^]'.
220 goldorak.bla.fr ESMTP
EHLO bubble.esip.univ-poitiers.fr
250-goldorak.bla.fr
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-STARTTLS
250 8BITMIME
MAIL FROM: ygritte@univ-poitiers.fr
250 Ok
RCPT TO: jonsnow@free.fr
250 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
From: ygritte@north.fr
To: jonsnow@north.fr
Subject: What do you know ?

You know nothing.
.
250 Ok: queued as 6D766293
```

Il n'est pas nécessaire de se connecter au serveur SMTP du domaine du destinataire. La particularité de SMTP est l'acheminement de machine en machine jusqu'à la destination. Notons que le serveur SMTP considère les en-têtes du mail comme des données du message (`From`, `To` et `Subject` dans notre exemple). Les en-têtes doivent être séparés du corps du message par une ligne blanche et ce dernier doit se terminer par une ligne comportant uniquement un point. Le protocole SMTP ne réalise aucun contrôle sur le contenu de l'en-tête. En revanche, il a connaissance de l'expéditeur et du destinataire précisés par les commandes *Rcpt to* et *Mail from*. Un serveur SMTP peut ou non choisir de relayer les mails selon la politique suivie par l'administrateur. Il peut filtrer les mails en fonction de l'adresse IP de la machine cliente, des renseignements donnés dans *Mail From* et *Rcpt to* etc... Cette politique de filtrage, plus ou moins drastique a été rendue nécessaire par l'augmentation continuelle de messages non-sollicités (Spam).

4 Domain Name System

Le DNS est une sorte d'annuaire permettant d'associer une adresse IP à un nom de domaine. Les noms de domaine constituent une arborescence. Les domaines de premier niveau (TLD) peuvent être de type *pays* ccTLD sur 2 lettres ou *générique* gTLD sur au moins 3 lettres. Les serveurs ayant autorité sur les TLD sont connues des DNS racines (`a.root-servers.net` à `m.root-servers.net`, mais correspondant à plus d'une centaine de machines (figure 1), ou voir root-servers.org[1])

L'annuaire fonctionne par délégation. Chaque domaine contient la liste des DNS pour chacun de ses sous-domaines.

Les requêtes peuvent être récursives (dans ce cas le DNS interrogé fournit la réponse en ayant lui même éventuellement effectué des requêtes DNS) ou itératives (dans ce cas le DNS interrogé fournit comme réponse l'adresse d'un DNS qui pourra répondre, à la charge du client de faire itérativement la nouvelle requête (voir figure 2). Les deux modes peuvent cohabiter.

[1] https://root-servers.org/

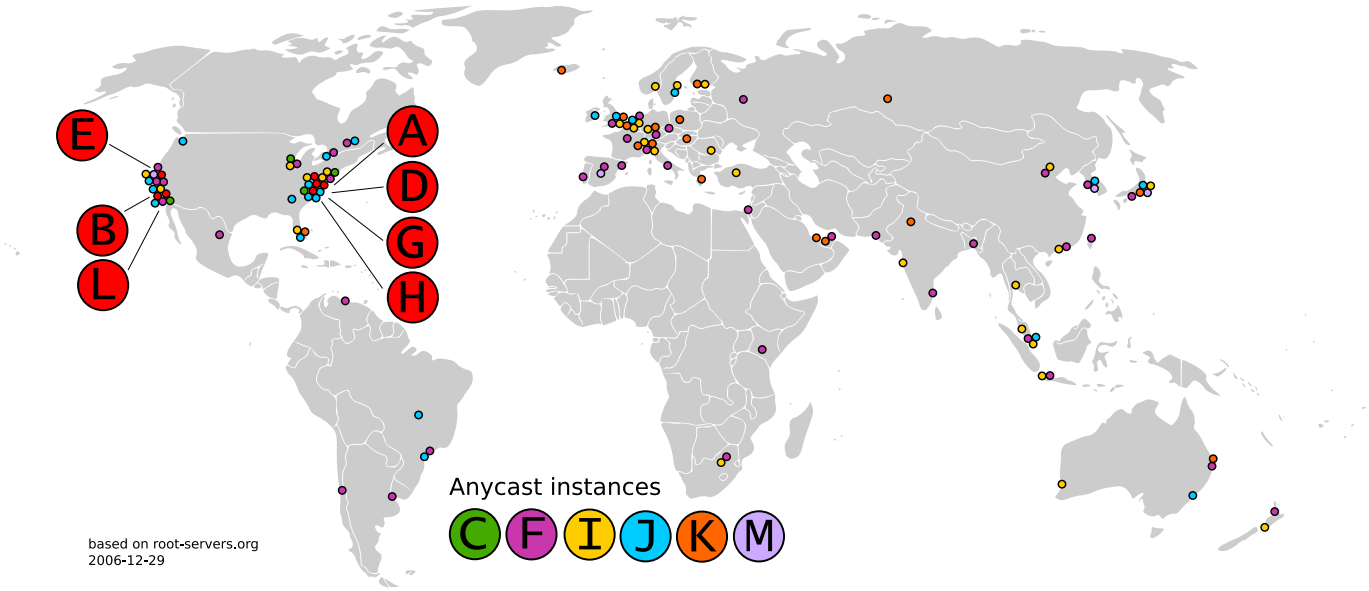

Figure 1: Serveurs DNS racine (Wikimedia Root-currenty.svg by . Matthäus Wander)

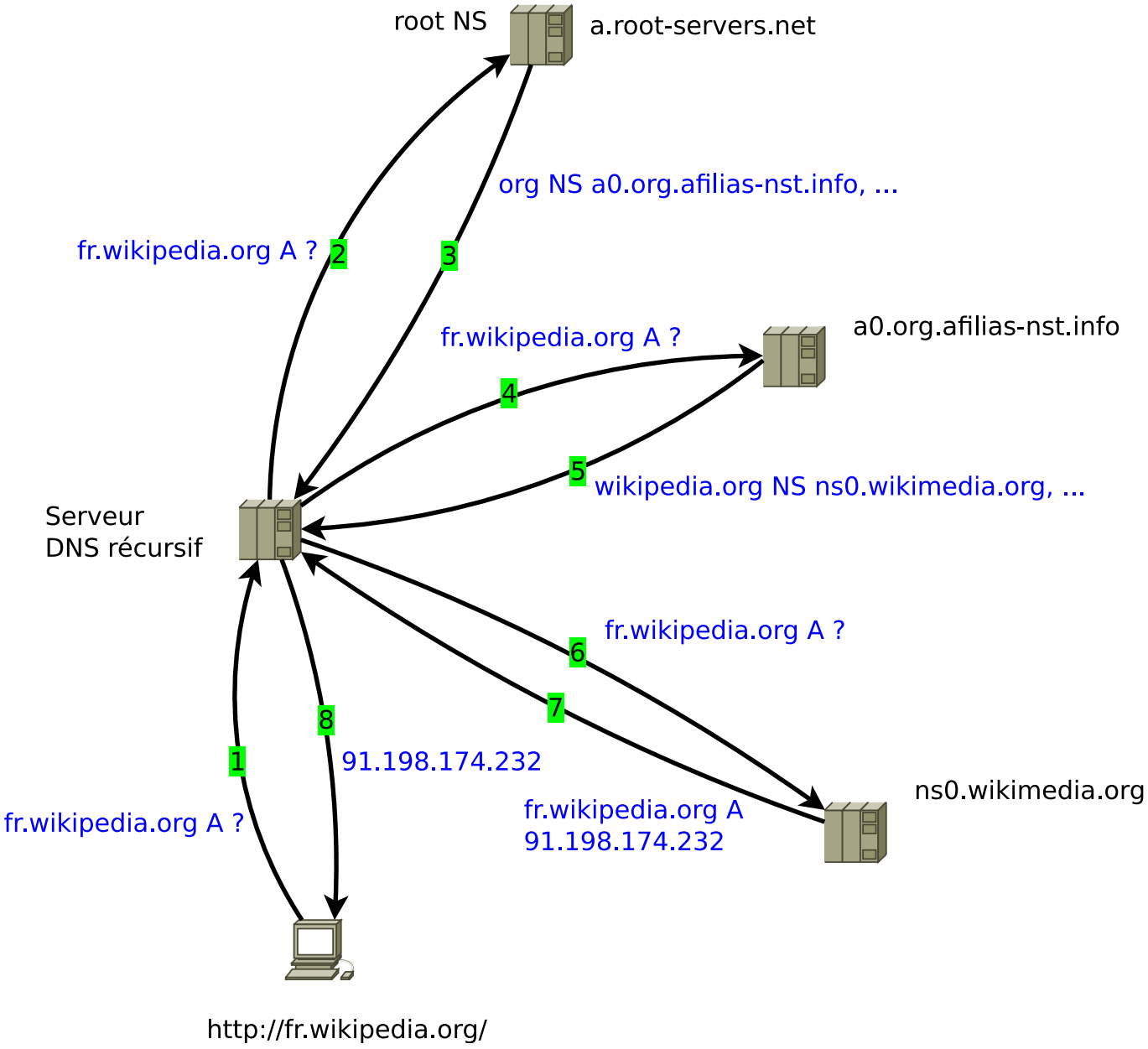

Figure 2: Résolution du nom `fr.wikipedia.org`, by Mro CC BY-SA 3.0

Les serveurs récursifs peuvent éventuellement mettre les réponses en cache (pendant un temps maximum déterminé). Un serveur DNS qui a délégation sur un sous-domaine fait *autorité* sur ce sous-domaine (inversement, la réponse *non-authoritative answer* d'un serveur indique qu'il a obtenu la réponse, mais ne fait pas autorité sur le sous-domaine).

En plus des enregistrements donnant les adresses en fonction des noms, le DNS peut fournir d'autres informations, comme l'adresse IP du serveur de courrier associé à un domaine.

Techniquement, DNS est décrit par plusieurs RFC (RFC 8499[2] pour une récente), il utilise UDP (bien qu'il ait aussi la possibilité d'utiliser TCP) et le port 53.

5 Dynamic Host Configuration Protocol

DHCP (Dynamic Host Configuration Protocol) est utilisé essentiellement au démarrage des machines clientes et leur permet d'obtenir automatiquement une configuration TCP/IP, sans intervention de l'utilisateur à condition qu'un serveur DHCP soit présent dans le réseau local.

Le client génère un message DHCPDISCOVER, contenant un identifiant (l'adresse MAC par exemple) en utilisant le protocole de transport UDP, à destination de l'adresse IP 255.255.255.255, et du port 67. Seul le serveur va répondre à cette requête en envoyant un message DHCPOFFER, vers le port 68 de toutes les machines de son réseau IP (255.255.255.255), L'identifiant étant renvoyé dans ce message, le client DHCP sait que le message lui est adressé. Ce message contient la configuration que pourrait prendre la machine (adresse IP, masque etc.).

Le client reçoit potentiellement plusieurs offres (il peut y avoir plusieurs serviers DHCP). Il répond avec un message DHCPREQUEST indiquant quelle offre il accepte. Le serveur dont l'offre est acceptée confirme avec un message DHCPACK.

6 Message Queuing Telemetry Transport

MQTT est un protocole de messagerie basé sur TCP sur le modèle *Publish/Subscribe*. Il est en particulier utilisé dans la communication des objets connectés.

Un serveur MQTT (appelé *broker*) récolte les messages des clients qui publient (*publish*) et renvoie ces messages aux client qui y ont souscrit (*subscribe*). Les messages sont organisés dans des *topics* arborescents.

Par exemple, un client sonde de température dans le salon de MrX pourrait publier dans le topic : `MrX/salon/sonde01`. Il enverrait régulièrement un message contenant la température mesurée.

Un autre client (par exemple le système de régulation du chauffage) pourrait avoir souscrit aux messages du topic `MrX/salon/sonde01`. Il recevrait alors les informations de température au moment où elles sont publiées.

MQTT utilise les numéros de ports suivants : 1883, 8883 (SSL).

Il existe plusieurs logiciels serveurs et clients (un des plus utilisés est mosquitto[3]), ainsi que des bibliothèques pour la plupart des langages de programmation (par exemple Paho[4])

On trouve sur internet plusieurs serveurs MQTT publics, utilisables essentiellement à des fins de test (une liste est disponible sur github[5]).

Pour se mettre en écoute d'un topic avec Mosquitto :

```
mosquitto_sub -h nom.serveur.mqtt -t MrX/salon/sonde01
```

Pour publier sur un topic avec Mosquitto :

[2] https://tools.ietf.org/html/rfc8499
[3] https://mosquitto.org/
[4] https://www.eclipse.org/paho/downloads.php
[5] https://github.com/mqtt/mqtt.github.io/wiki/public_brokers

```
mosquitto_pub -h nom.serveur.mqtt -t MrX/salon/sonde01 -m '18.7'
```

Un système de caractères jokers permet à un client de souscrire à plusieurs topics, ou à un arborescence complète. Outre la possibilité de chiffrement, MQTT peut être configuré pour nécessiter une authentification.

7 Mémo outils de connexion

On dispose de quelques outils pour expérimenter les connexions réseau, et se substituer à un protocole application.

En ligne de commande, les outils `telnet` et `nc` permettent de facilement simuler une connexion TCP en mode texte non chiffrée :

```
telnet mon-site-web.fr 80
nc mon-site-web.fr 80
```

L'outil `telnet` est disponible aussi sous Windows. Sur cette plateforme, on pourra utiliser un outil plus sympathique comme PuTTY.

Dans le cas des services chiffrés, la partie chiffrement peut être réalisée par `openssl`. Du point de vu de l'utilisateur, on travaille comme avec une connexion non chiffrée :

```
openssl s_client -connect mon-site-web.fr:443 -crlf -quiet
```